Beowulf Database Option

Robert Schaefer
GLAST SSC
Database Options

- DBMS (PostgreSQL, Oracle, Sybase, etc.)
- Beowulf
- Other options not thought of yet…
Outline

• A little about Beowulf Clusters
• Simple architecture used in prototype
• Performance
• Improvements
• Conclusions
Beowulf Cluster

• Cluster of low cost mass market computers
• Tied together with a message passing interface (usually MPI or PVM) and synchronization/management tools
• Memory is not shared.
• Pioneered by people at GSFC
How Beowulf Would Work

• Effective parallelization means that the photons are sorted randomly.
• If spatial searches are most important, can store them in time order (or use a hash function).
• No need to build and maintain a large index, just store metadata.
Simple Architecture Prototype

- Original FITS files from level 1 processing kept on disk
- Ingest of new data - copy new file to disk
- Reprocessing - replace FITS file on disk
Simple Beowulf Photon DB

Server on Master

MPI / PVM

Nth Slave processor

Photon query

1st Slave processor

RAID

Photon file

Photon file

Photon file

RAID

Photon file
Simple Performance Tests

- Prototype architecture
  - Query all photons in a region
  - Master tells slaves which files to search
  - Slaves search files, make FITS file of queried photons.
  - Slaves send resultant files back to master.

- Tested searching boxes in galactic coordinates using simulated GLAST data (Seth Digel)
Prototype Performance Times

Beowulf performance

- 15x15, 100 d
- 30x30, 100 d
- 45 x 45, 100 d
- 15 x 15, 300 d
- 30 x 30, 300 d

Number of processors vs. time per photon searched (microseconds).
Prototype Meets Requirements

• Searches can be done at \( \sim 1 \) microsecond per photon.
• With 1 G photon database (after 10 years), search can be done in 1000 s = 17 min.
• Requirement is \( \leq 30 \) min./year data searched.
Performance Scaling

Beowulf performance

- 15x15, 100 d
- 30x30, 100d
- 45 x 45, 100 d
- 15 x 15, 300 d
- 30 x 30, 300 d
- no F T, 100 d
- no F T, 300 d
How to Improve Performance

• 1st bottleneck is inefficient file transfer to master
  – Gigabit ethernet
  – more efficient file transfer algorithm
• 2nd bottleneck is disk access times
  – Store photons in memory
  – 40 nodes w/ 5 Gb gives 200 Gb storage.
• Could possibly get a factor of ~ 30 improvement
  – Entire database searched in 30 seconds.
Improved Beowulf Photon DB

Server on Master

MPI / PVM

Photon file

RAID

Photon query

1st Slave processor

Nth Slave processor

... γ in memory

γ in memory

RAID

Photon file
### Beowulf vs. DBMS

<table>
<thead>
<tr>
<th>DB Attribute</th>
<th>Beowulf</th>
<th>DBMS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Querying Times</td>
<td>No query intrinsically longer than another</td>
<td>Spatial queries fast, but complicated joins may be long</td>
</tr>
<tr>
<td>Reprocessing / Update Times</td>
<td>Easy replace/add file, read new file</td>
<td>Find and delete old photons, ingest new</td>
</tr>
<tr>
<td>Programming Times</td>
<td>Requires Beowulf programming up front</td>
<td>Uses COTS</td>
</tr>
</tbody>
</table>
Conclusions

- Beowulf can meet our needs and much more
- Beowulf database architecture is simple, but all code must be custom.
- DBMS can likely meet our needs as well.
- Must choose architecture makes the most sense.